FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Game Design Patterns in Serious Games for Software Engineering Education

João Nogueira

DISSERTAÇÃO

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Nuno Flores

August 1, 2018

Game Design Patterns in Serious Games for Software Engineering Education

João Nogueira

Mestrado Integrado em Engenharia Informática e Computação

August 1, 2018

Abstract

Most of the gaming industry aims to satisfy and entertain a user or group of users with their gaming experience and to keep them as engaged as possible in the piece of software developed. However, there are games with a different goal. These are called Serious Games. They are especially designed with a primary goal different from pure entertainment. We could classify a flight simulator, used in pilot's training, as a serious game. However, in this work we will be referring to serious games in the education area, how they can be used to teach and how can we ensure that they are as effective as possible in teaching Software Engineering students. Games have a lot of potential in this area as they provide an environment where the student can learn in a practical manner while being able to control and provide support throughout the learning process.

When talking about entertainment aimed games, we should consider that they are usually developed focusing on satisfying the player and keeping him as engaged as possible, however, in this case we need to consider how much he is learning. Keeping the scale between learning outcome and engagement balanced is a difficult task [BÖBH16], and the way the game is designed has an effect on both ends of this balance.

There is already a wide range of serious games aimed at teaching areas in software engineering. Among those games there are some relying on different game design patterns, however, the relation between these patterns and the player's engagement and learning outcome is not clear yet. In order to better understand how a serious game can be as effective as possible in teaching Software Engineering, we first need to understand how these game design patterns relate to a positive teaching result.

This work aims to clarify the weight of different game design patterns in serious games developed with the aim of teaching software engineering topics through an extensive survey of the existing serious games and respective game design patterns as well as possible further experiments.

Resumo

A maior parte da indústria de jogos tem como objetivo satisfazer e entreter o utilizador ou grupo de utilizadores com a sua experiência de jogo e mantê-los o mais dedicados possível com o produto desenvolvido. No entanto, existem jogos com um objetivo um pouco diferente. São estes os jogos sérios. Estes são especialmente desenvolvidos com o objetivo primário diferente de puro entretenimento. Podemos classificar um simulador de voo, utilizado no treino de pilotos, como um jogo sério. De qualquer forma, neste trabalho vamos referir-nos a jogos sérios na área da educação, como podem ser utilizados para ensinar e como podemos garantir que estes são tão eficazes quanto possível no ensino de estudantes de Engenharia de Software. Jogos têm muito potencial nesta área pois proporcionam um ambiente onde o estudante pode aprender de forma prática enquanto se mantém o controlo e se disponibiliza apoio ao ensino por todo o processo de aprendizagem.

Quando se desenvolve jogos destinados ao entretenimento, normalmente desenvolve-se o jogo focando-nos na satisfação do jogador e em mantê-lo o mais dedicado possível, no entanto, neste caso em específico é necessário ter em consideração o quanto o jogador está a aprender. Manter o equilíbrio entre o resultado de aprendizagem e o nível de dedicação do jogador é uma tarefa complicada [BÖBH16], e a forma como o jogo é desenvolvido tem um grande impacto em ambos os lados da balança.

Neste momento, já existe um grande leque de jogos sérios destinados ao ensino de Engenharia de Software. Estes mesmos jogos apoiam-se em diferentes padrões de desenho, no entanto, a relação entre os padrões de desenho utilizados e o nível de dedicação do jogador e o resultado de aprendizagem ainda não é clara. Por forma a compreender melhor como é que um jogo sério pode ser tão eficaz quanto possível no ensino de Engenharia de Software, é necessário compreender como os padrões de desenho de jogos se relacionam com um resultado positivo no ensino.

Esta dissertação tem como objetivo clarificar o peso dos diferentes padrões de desenho no desenvolvimento de jogos sérios para o ensino de Engenharia de Software através de uma pesquisa extensiva dos jogos sérios existentes e respetivos padrões de desenho, bem como possíveis experiências futuras.

Acknowledgements

This thesis is the final step in a long journey. The final step of my time as a student. First of all, I owe the opportunity to study and grow up as I did to both my parents, who made sure I had such an opportunity. I could never be more grateful to them. To my sister, who put up with me for all these years and will keep doing so, for the years to come.

A big thank you to my childhood friends, who keep being so even though we don't see each other for long periods of time. To those who made this journey with me, since 2013, who made this journey as legendary as possible. I hope that those who come after can have people like you by their side.

To my thesis' supervisor, Nuno Flores, who helped me through every step of this dissertation, a word of appreciation for his effort and help throughout this process.

João Nogueira

"Wanting something does not give you the right to have it."

Ezio Auditore, Assassin's Creed 2

Contents

List of Figures

LIST OF FIGURES

List of Tables

Abbreviations

GDP	Game Design Pattern
LTF	Teaching and Learning Function
SE	Software Engineering
SEE	Software Engineering Education
KA	Knowledge Area
SWEBOK	Software Engineering Body of Knowledge

Chapter 1

Introduction

Serious Games are a concept most people are not familiar with, but most of them have already interacted with some kind of Serious Game throughout their life. A video game is defined as any kind of software that has as its primary goal the entertainment of the user. However, some video games have emerged as having other kinds of goals. Nowadays, several video games have been released not having entertainment as their primary goal, and having, in it's place, teaching or improving a topic or skill.

> "[...] we will rely on a broader definition of "Serious Games": any piece of software that merges a non-entertaining purpose (serious) with a video game structure (game)" [DAJ11]

Considering this definition, we can include many different kinds of software in the large group of Serious Games. In this particular case, we will focus on any kind of game developed as having the teaching of Software Engineering as its primary goal, and the relations between the Game Design Patterns used in the video games and their teaching outcomes.

1.1 Context

As soon as games started to be used as a tool to teach Software Engineering the need of understanding what makes them effective or not became of most importance. As it happens in any other industry, a tool that helps you understand what makes a product effective in achieving its goal is very valuable for any kind of developer or academic.

Many aspects of a game contribute to the final outcome of a game, specificaly in this case, to the teaching outcomes of a game. This work is foccused on the outcomes of a major characteristic of games: gameplay.

Before being able to establish the relation between the game and the teaching outcomes, we need to have a tool to describe and analyse each game's gameplay, in order to have a different

set of characteristics to which the success of the game, or otherwise, will be associated. In his book, Björk introduces Game Design Patterns (GDPs) "as a way to implicitly state what a game is" [BH05, chapter 3], providing a tool to collect the characteristics needed to this study.

Although some work in this area already exists [LPF15] [Far16], a clear relation between these Game Design Patterns and their teaching outcomes in Software Engineering Education (SEE) does not exist yet.

1.2 Motivation

Software Engineering is a very broad area of studies that impacts not only the way different companies or individual developers deal with software but also how we think of software. When thinking of software development we no longer think of a single final product but we think in the concept, design, development and maintenance required.

The increase of students interested in the area justifies the increase of the interest in the various different ways of teaching. Serious Games are becoming a reality in teaching this area, however, the full potential of this tool in teaching is yet to be reached, which would change if a better understanding of the perfect balance between the GDPs and the teaching outcome existed.

Even though the process of learning is viewed by many as a simple relation between receiving new information and remembering it, Grosser argues that the process of learning is much more complex and requires a series of complex functions between the transmission of the new information by the teacher and assimilation of the new information by the learner [Grö07], having the support of specialists in the area. Linking these different learning functions with the GDP is a necessary step to relate these patterns with the learning outcome.

Any kind of game has, by definition, an entertainment part which ensures the engagement of the player. Combining the player's engagement in the game with the maximum possible learning outcome is paramount to the developer of the serious game.

1.3 Goals

As previously stated, this dissertation aims to find the relation between the GDPs that are used in the development of Serious Games to the education of Software Engineering and the learning outcome of these games. Related studies already exist relating the Learning and Teaching Functions (LTFs) and the GDPs considered important to the teaching process of Software Engineering Management [LPF15] and Software Requirements [Far16].

Building on the findings of the previous studies, this dissertation aims to consolidate and expand the knowledge on the relation between LTFs and GDPs, using a different approach than the one Pedro Letra used to get to the subset of GDPs.

Another goal of this dissertation is to have, by the end, a guide that clearly states which GDPs are directly related to different learning outcomes and how their relation works, so that in the

future, Serious Games developers can assess in advance the impact that their games can have in teaching Software Engineering.

1.4 Structure

This document is structured in six different chapters.

The present chapter introduced the work that was done, as well as the main goals of the dissertation. It is also emphasized why this study is important and who the most interested parties are.

The following three chapters contain all the concepts and previous studies needed to know to understand the work done in this dissertation. In the second chapter, named "Software Engineering Education", concepts like Software Engineering and SWEBOK are introduced whereas the third chapter focuses on Serious Games and breaking down Games in different Game Design Patterns, defining what a pattern is and how we identify it in a game. The fourth chapter regards related work done in this field of study trying to relate Game Design Patterns with learning outcomes of the player.

The fifth chapter of this dissertation is reserved to the survey itself. A description of the game selection method can be found,r as well as the results of the survey and its discussion.

Finally, in the sixth chapter we have the dissertation's contributions to the community, conclusions from this work and some suggestions of future work that might be done built on this dissertation.

Chapter 2

Software Engineering Education

Before developing the study itself an introduction of the most relevant concepts related to this dissertation needed to be made. As such, in this chapter we find the description of some of these concepts related to Software Engineering (SE) and to the challenges of teaching this field.

2.1 Knowledge Areas of Software Engineering

Considering the topic in which this dissertation is inserted, it is of most importance to define and understand the concept of Software Engineering. The International Organization of Standardization defines Software Engineering as:

> "the systematic application of scientific and technological knowledge, methods, and experience to the design, implementation, testing, and documentation of software to optimize its production, support, and quality" [BAB+06]

This means that whenever anyone refers to Software Engineering, that person is not referring only to the final product or the idea, but to any of the parts that compose the complex process that is Software Engineering.

In the area of Software Engineering, fifteen different Areas of Knowledge (KA) can be identified according to the Software Engineering Body of Knowledge (SWEBOK) [BF14] as shown below.

- Software Requirements

  This KA is related to the elicitation, analysis and validation of the requirements, as well as to the management of the dynamic property of these requirements as they may change throughout the software development process.

- Software Design

The KA called Software Design is defined by The Joint Task Force on Computing Curricula IEEE Computer Society Association for Computing Machinery as being "the process of defining the architecture, components, interfaces, and other characteristics of a system or component" [BF14] and the result of this process. This is considered to be the life cycle activity in which the software's internal structure is produced by analysing the Software Requirements.

- Software Construction

  This section refers to an KA that is strongly linked to all other KAs. However, one can argue that the strongest link of this specific KA is to Software Design and Testing as it refers to the coding, verification and unit-testing of the software it is related to.

- Software Testing

  The name of this KA is self-explanatory. It refers to a continuous verification that the implemented code fulfils the requirements set either by using unit-testing or other kinds of more complex tests.

- Software Maintenance

  Software maintenance is an activity needed not only after the first implementation but also during and after the implementation itself. Even knowing that the requirements are set in the beginning, the environment around the product may have some changes. The requirements may also change or evolve over time which means that Software Maintenance is an essential KA for SE.

- Software Configuration Management

  In its definition, it is stated that:

  > "A system can be defined as the combination of interacting elements organized to achieve one or more stated purposes. The configuration of a system is the functional and physical characteristics of hardware or software as set forth in technical documentation or achieved in a product" [BF14]

  Which means that the software configuration can also be thought of as the configuration of a collection of different firmware, hardware or software that need to be combined to achieve a specific purpose.

- Software Engineering Management

  The process of managing the different components of a project is a KA in itself. The managing activities required to ensure an efficient and satisfying final product. "Planning, coordinating, measuring, monitoring, controlling, and reporting" [BF14] are some of the activities that are considered to be a part of this AK.

- Software Engineering Process

  In order to achieve certain outputs, a group of different activities that relate to each other is required, transforming a number of inputs into that requested output. This is a process that also requires consuming certain resources and it is called an Engineering Process.

- Software Engineering Models and Methods

  As in any kind of project, a certain organization is required to ensure the best outcome is achieved. By imposing certain Models and Methods to the work being done in a SE project, it is possible to ensure that what is being accomplished is repeatable and structured in an understandable way allowing the project to succeed.

- Software Quality

  This term, although very common, refers to the importance of ensuring the several desired characteristics of software products and to different kinds of processes, techniques and tools that can be used to ensure that these required characteristics are met [BF14].

- Software Engineering Professional Practice

  This particular KA is concerned with several attributes that the Software Engineer must have in order to ensure a responsible, professional and ethical way of work. Some of these attributes are different skills, areas of knowledge, attitudes or even behaviours in the work-place or towards the work that the engineer has.

- Software Engineering Economics

  As in any other industry, Software Engineering projects require a complex business structure and it's proper management is of most importance. This KA is concerned with this kind of management.

- Computing Foundations

  This KA is related to the environment in which any software product is developed and subsequently executed: The Computer.

  > "Because no software can exist in a vacuum or run without a computer, the core of such an environment is the computer and its various components." [BF14]

  As suggested by the quote above, this KA is related to the main principles and rules that govern computing.

- Mathematical Foundations

  Mathematics is much more than solving simple arithmetic problems and dealing with numbers. The ability to understand the logic and reasoning is the main subject on which this KA is turned to. Understanding this concepts and how they are translated to any coding language is an important asset to any Software Engineer.

- Engineering Foundations

  This KA simply refers to the basic skills and areas of knowledge inherent to any kind of Engineer. These skills are important to the success of the Software Engineering project, as in any other kind of engineering project.

2.2 Issues of Teaching of Software Engineering

As mentioned above, Software Engineering is an area composed of many different knowledge areas that need to be understood and correctly related by the engineer to ensure that the the product is correctly and fully developed. However, it is not enough for these areas to be taught passively through lectures and classes.

Nowadays, apart from this passive teaching technique, Software Engineering is taught by simulating an environment as close to the one the Engineer will find in the industry by creating conditions similar to the ones found in a company. However, developing a downsized project in a classroom environment is not enough to ensure that the student is ready for the real challenge [Tch11].

Software Engineering is a very broad concept and not an exact process. What this means is that many problems are dependent on the surrounding environment and many different conditions that may occur before, during or after the development of a specific project.

The teaching of this area was tackled as most of the areas in college, with lectures, software or presentation materials, to allow the student to learn the concepts and with some, but limited, in-class practise using some small scale projects. Even though some exceptions exist, most of the classes still rely on these techiques, as shown, for example, in the *MEng Computing (Software Engineering)* course of the Imperial College of London [Imp] and in the *Software Engineering 2* module of the *BSc Software Engineering* of Manchester University [Man]. Some different techniques are used but ultimately, the teaching of SE still heavily relies on those more traditional methods. This method has been proven somewhat inefficient, or at least, not as efficient as they need to be to meet the industry's requirements, as Emily Oh acknowledges:

> "The software engineering industry is still noting a large disparity between the software engineering skills taught at a typical university or college and the skills that are desired of a software engineer by a typical software development organization" [Oh02]

Oh found five major issues that the current methods of teaching Software Engineering cannot fully tackle: SE is not linear, often multiple different and conflicting goals are present, the choice from many different alternatives is required, the existence of many stakeholders and may exhibit dramatic consequences.

Some solutions to these teaching issues have been attempted and implemented, such as Problem Based Learning, as Santos states:

> “A learning environment, though, should not only be practical but also true to the market reality. Problem-based learning (PBL) is an appropriate way of doing this, being focused on putting students at the center of the learning process and involving them in real situations." [dS17]

Problem based learning is one of the methods found to tackle the complex problems surrounding the teaching of SE. A specific approach of Problem-based learning is the use of Open Sourced projects as a teaching tool. Students are involved in these projects, actively working on a project that is not simulating a real-world environment but is, instead a real-world project.

> "Students in FLOSS projects have the chance to interact with field practitioners and gain valuable experience on the domain." [PSM13]

Being an integral part of the industry, this approach is one way of tackling some of the challenges of teaching directly related to the fact that teaching lacks some of the real world projects' characteristics. However, Serious Games are particularly effective in tackling the five issues previously named, as shown below [Oh02]:

- SE is not linear - SE is a process that adapts to a projects specific needs making it different for each project. Knowing that most games include some factor of randomness, any run of a game can be rendered unique;
- Often multiple different and conflicting goals are present - it is frequent for the engineer to be presented with a situation in which he must choose between conflicting goals. Games simulate this situation very often, having *Conflicting Goals* as one of their Design Patterns;
- The choice from many different alternatives is required - Games allow players to choose what action to take and allow them to choose more than once for each choice, as a save-load cycle normally exists;
- The existence of many stakeholders - mostly simulated in multi-player games as in these games each player competes for the best individual performance;
- May exhibit dramatic consequences - In most games, the outcome of the narrative is influenced by the player's choices, even if they result in a radical plot change.

2.3 Summary

In this chapter the concept of Software Engineering was introduced as well as the fifteen different Areas of Knowledge within SE identified in SWEBOK. These KAs are important for this work as they were relevant to the selection process of the serious games that were analysed.

Regarding the act of teaching SE and the specific KAs of SE, it is important to remember that many of them are still taught in an "old-school" lecture like manner, or using small scale projects with real life characteristics, which have some major issues. Some of these issues can be tackled with the teaching of SE using Serious Games.

Chapter 3

Designing (Serious) Games using Patterns

This chapter we will introduce the concept of Serious Games and the framework used to analyse such games.

3.1 Concepts

This thesis is, as its name suggests, deeply related with Serious Games for Software Engineering Education. As such, it is only fitting that we define what a Game is, how it becomes a Serious Game and the specifics of Serious Games for SEE.

- **Games**

  Many philosophers and academics have tried to define a game, and many of them have come up with different definitions for a game. However, one of the definitions considered most accurate was made by Salen and it is as follows:

  > "A *game* is a system in which players engage in an artificial conflict, defined by rules, that results in a quantifiable outcome" [SZ04]

  Even though this definition is accurate for most cases, some products that are considered games are not necessarily based in conflict but contain a set of rules that the players must respect and they also contain a set of goals, even if they are not related at all with overcoming an enemy on any kind of conflict. As Salen says, Conflict can be considered a contest of powers [SZ04], which means that the player is always, somehow, competing to achieve a determined goal.

- **Serious Games**

Having defined what a game in general is, the need to understand what makes any game a serious game arises.

One of the first definitions of Serious Game that can be found was made by Abt who describes a Serious Game as a simulation or game to improve education. Which means that any game developed with the aim of teaching or improving any skill can be considered a Serious Game [Abt87]. It is important to note that, in any kind of Serious Game, its main objective is not the entertainment of the player, as it is primarily focused on teaching or helping the player acquire a certain skill.

- **Serious Games (video-games)**

  One important definition that appeared after the introduction of the video-game concept was made by Sawyer in 2002 who said that a Serious Game relies on the connection of the technology and knowledge of the video-game industry and a determined serious purpose [Saw03].

  Some studies have been conducted after this definition appeared, and many academics have a somehow different opinion on what a Serious Game is. One of the reasons that makes the boundaries of this definition quite hard to define, is the fact that many different academics from many different areas of study have different opinions on its definition [DAJ11]. However, one of the definitions agreed upon by most specialists is:

  > "[...] broader definition of “Serious Games”: any piece of software that merges a non-entertaining purpose (serious) with a video game structure (game)." [DAJ11]

  Considering the quote above, a Serious Game has an entertainment purpose and a non-entertainment one. This non-entertainment purpose can be teaching, either in the Software Engineering area or any other, improving a skill, giving relevant information to the player, for example, to serve as a tutorial for a task in the real world, or any other non-entertainment purpose.

  In this dissertation, a method to define what is a Serious Game for SE Education will be defined. However, this method will rely on this broader definition of what a Serious Game is.

3.2 Serious Games for Education

The process of learning is not always motivating for the student. This is an issue that Serious Games can tackle. According to Prensky [Pre01], games are the most engaging pastime in history mainly due to the following reasons:

- Games are fun providing the player with a form of pleasure;
- Games are a form of play making the player feel involved;

- Games have rules making them structured;
- Games have goals included that motivate the player to keep trying to overcome them;
- Games are interactive providing the player with something to do;
- Games are adaptive;
- Games have outcomes and provide feedback providing the main tools for a learning process;
- Games have win states making the player feel or try to feel gratified;
- Games have conflict, competition, challenge and opposition, even if only by the AI, providing adrenaline;
- Games have problem solving, many times requiring the player to become creative about his solutions;
- Games have interaction allowing the player to be social;
- Games have representation and story adding emotion to the player's play time.

Taking these reasons into account, it seems only fitting to mix games with the learning process allowing the student to learn what is being taught while feeling engaged and wanting to keep playing/learning.

Games provide a teaching environment that cannot be provided in any other way, as Kirriemuir asserts:

> "The instant feedback and risk-free environment invite exploration and experimentation, stimulating curiosity, discovery learning and perseverance" [Kir02]

Video-games have also been found to provide significant other benefits in teaching. In a study made by Mitchell and SavillSmith, some of these benefits are mentioned as follows:

- Improved strategic thinking and insight;
- Better psychomotor skills;
- Development of analytical and spatial skills;
- Visual selective attention;
- Computer skills.

Further advantages of using video games as a teaching tool include, when referring to expert players, the following expert behaviours [MSS04]:

- Self-monitoring;

- Pattern recognition;
- Problem recognition associated with problem solving at a deep level;
- Principled decision making;
- Qualitative thinking;
- Good short and long term memory.

3.3 Game Design Patterns

In order to analyse a game, there is a need to break it down into different pieces allowing us to compare different games and describe what each of them has, or not, as feature. To do so, we need to define what a Game Design Pattern is. Game Design Patterns have been defined by Bjork as follows:

> "game design patterns are semiformal interdependent descriptions of commonly re-occurring parts of the design of a game that concern gameplay [BH05].

A GDP is semiformal because there is no formal way to represent it. Most GDPs get noticed during gameplay but have completely different implementations which means that a formal definition is unreachable. Some GDPs are interdependent because the existence of one GDP mandates, many times, that other GDPs occur, for example, if a game has *competition*, then it is only logical that *conflict* also exists within the game.

While defining the collection of game design patterns Bjork divided the GDPs in the following categories:

- Game Design Patterns for Game Elements;
- Game Design Patterns for Resource and Resource Management;
- Game Design Patterns for Information, Communication, and Presentation;
- Actions and Events Patterns;
- Game Design Patterns for Narrative Structures, Predictability, and Immersion Patterns;
- Game Design Patterns for Social Interaction;
- Game Design Patterns for Goals;
- Game Design Patterns for Game Sessions;
- Game Design Patterns for Game Mastery and Balancing;
- Game Design Patterns for Meta Games, Replayability, and Learning Curves.

Each of the GDP categories found above has several GDPs and sub-categories. The complete framework can be found in the appendix A, at the end of this document. This is the definition and collection of Game Design Patters that we will rely on for this dissertation.

3.3.1 Game components - Analysis framework

As in any other area of study, a standardization or framework is required to be able do communicate what is intended. To allow academics to better describe what they observe, and discuss their findings and studies about games and their gameplay, Bjork came up with a framework that divides the game components into four different categories.

3.3.1.1 Holistic component

> "The holistic components deal with the aspects of a game that are relevant when one looks upon the activity of playing games as an undividable activity" [BH05]

This way of looking at games is important when trying to relate the player's activities while playing and while not playing a game. Some concepts where introduced, four of which can be found below:

- **Game instance**

  The game instance refers to the whole lifetime of a game, from it's beginning to it's end.

- **Game Session**

  The game session refers to a complete session of a single player. Since the player sets up his initial game session and starts playing until he stops playing. A Game instance may include several game sessions, however, the reverse is not possible.

- **Play Session**

  Even though a game session describes the complete game of a single player, in some games the player may stop playing, leave the system and then, eventually, resume the game. Both times the player plays the game are considered to be of the same game session but of two different play sessions, which means that a game session may be composed of several play sessions but a play session belongs to a single game session.

- **Extra-game Activities**

  The Extra-game activities component refers to all activities that the player may have that are related to the game but not necessarily impact the game, such as sharing constructions on simulation games, for example.

3.3.1.2 Boundary component

> The boundary components are those that limit the activities of people playing games, either by only allowing certain actions or by making certain activities more rewarding." [BH05]

In a game, the player is supposed to have certain actions or activities that he is more or less compelled to do. Trusting that the player will follow the path that the game designer intends is not a possible way of thinking, since different players may have completely different visions and opinions on what the game means to them and of what they actually want to do in a game. Considering that, some game conditions were defined to limit the players decisions and divided into Rules, Modes of Play and Goals.

- **Rules**

  Rules are a way to limit the player's behaviour through the imposition of certain actions and forcing the player to have a limited set of actions. The sequence in which the player may act may also be described by the rules of the game.

- **Modes of Play**

  Many games are composed of more than one mode of play that allow players to have different sets of actions to choose from. A mode of play may be composed of a set of sub modes of play.

  The different modes of play can be explicit by providing different interfaces or implicit.

- **Goals and Subgoals**

  Goals are the main way to motivate players to thrive and fight to achieve a certain game state. Normally a goal is composed of many different subgoals. These goals can either be defined by the game or by the player. Some games do not impose goals to the player, allowing him to define his own goals while playing the game.

3.3.1.3 Temporal component

> "The temporal components describe the flow of the game, such as when telling someone else what took place in the game after the game has been finished." [BH05]

The temporal component is composed by actions that the player may have done or events that may have occurred in the game. The only thing these need to have in common is an impact on the game state itself.

The temporal component is divided in 5 different categories:

- **Actions**

  While playing the game a player may inflict changes on the game state of the game. The way the players can do this is through Actions.

- **Events**

  Even without an action by the player, the game state may change and this change is perceived in a certain way by the player. This is what is called an Event.

- **Closures**

  Closures are changes in the game state that have meaning to the player. Achieving goals or realising that a particular goal is not reachable are examples of closures to the player.

- **End Conditions**

  The change of a mode of play or end of game session has a set of conditions that need to be fulfilled to happen. These conditions are called end conditions.

- **Evaluation Functions**

  These are composed of algorithms and aim to evaluate the players performance after an end condition is met.

3.3.1.4 Structural component

> Structural components are the basic parts of the game manipulated by the players and the system" [BH05]

This is the component that is easier to identify in a game. It represents the real-world's or imaginary objects, characters or objects that exist in the game. This component is divided into five categories:

- **Game Facilitator**

  The game facilitator is some kind of agent controlling gameplay. In traditional games like Tag, have the players themselves as the game facilitators, however, in board games the board itself is the facilitator, and in video games, the software acts as the game's game facilitator.

- **Players**

  Players represent the different entities playing the game and working to achieve their own goals within the game's environment.

- **Interfaces**

  The interface can be defined as the method through which the players interact with the game. Without a good interface it is impossible for a game to achieve it's maximum potential.

- **Game Elements**

  Game elements are different objects or creations within a game that a player can manipulate in order to advance in a game and achieve his goals.

- **Game Time**

  Game Time is the sequence in which different actions performed by a player or events that the player perceives happen. Considering game time is defined as a sequence, it is independent of the game's real-time.

3.3.2 Identifying Game Design Patterns

In order to identify GDPs in a game, Bjork defined two different ways to analyse their presence. To do so, he relies mainly on the existence of the game itself, prototypes or game design documents. Apart from these, he also mentions the possibility of using instruction manuals or even code in order to obtain the broadest sources of information as possible ensuring the accuracy of the analysis.

That said, Bjork describes two very distinct ways of analysing a game and it's GDPs: Structural analysis and Play Testing. These forms of analysis have different requirements that are described below:

- Structural Analysis - The main focus of this type of analysis is to collect the set of GDPs that can be found to exist in a game without playing it using game design documents or other sources of information;

- Play Testing - Having someone play the game while analysing that person's gameplay. Note that for this analysis to be as accurate as possible, several analysis should be made with different players. Ideally, these players should not have any idea of what is being analysed, as this might make them choose different options or actions to take.

3.4 Summary

Serious Games have the potential of tackling some issues very closely related to the teaching of SE, as stated in the section 2.2.

Game Design Patterns and Bjork's components framework were explaind in this chapter. These are of most importance for the rest of this thesis as GDPs are a core concept of this work. The analysis of the games relied on these patterns and was based on the ways of identifying GDPs previously mentioned.

The concepts presented in this chapter are also required to fully understand the related work presented on the next chapter as well as it's conclusions and relation to the work developed in this thesis.

Chapter 4

Linking Game Design Patterns to Software Engineering Education

As in any kind of study, some research is always required or related studies to understand what already exists and how it was done. Concerning the subject of this dissertation, at least two different studies exist that try to link Game Design Patterns to the pedagogical outcome in Serious Games for Software Engineering Education.

Understanding the approach of these two different studies is paramount to the development of this work, allowing us to have a broad knowledge of what was already done in this field.

4.1 Software Engineering Management

In this study named *Game Design Techniques for Software Engineering Management Education* and carried out by Pedro Letra [LPF15], an attempt was made to find the relations between the Game Design Patterns and the teaching outcome of Software Engineering Management. In this section, the approach taken by this study will be introduced, as well as some essential concepts to help us understand the relationships between them.

In his work, Letra goes through the relationships represented in the triangle shown in the figure 4.1. The final relationship he finds, and the one he draws conclusions from, is the one represented by **c**. As an attempt to reach this relationship, the author works his way through the relationships **a** and **b** as described in the next sections of this chapter.

4.1.1 Learning and Teaching Functions

Many times it is considered that, in a learning environment, when someone teaches a certain subject, it is immediately learnt by the student. This is, without any doubt, an incorrect assumption. When that subject is not learnt by the student, the impulse is to blame their inability to focus or to learn that specific subject. Although that might also be possible, assuming that is the next assumption to make is incorrect [Grö07].

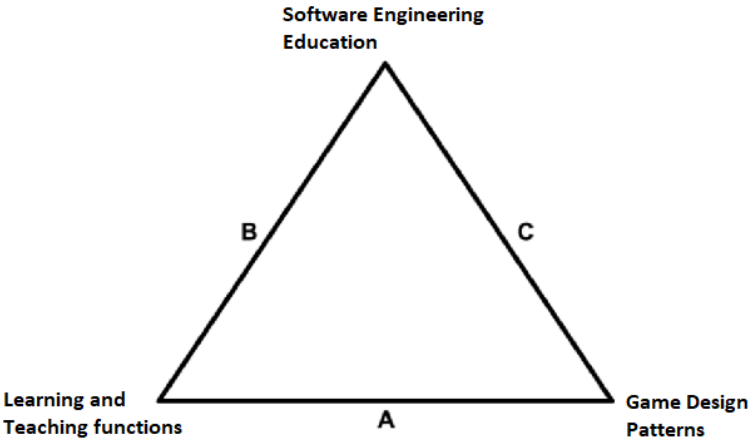

Figure 4.1: Concept Triangle - Approach model followed (adapted from [LPF15])

> "a teacher must teach not only content to learners, but also the functions required by the engagement with that content in order to make learning effective, meaningful, integrated and transferable" [Grö07]

Considering the quote above, the teacher needs to know the ultimate subject he wants to teach, but also, the teaching functions required to allow the learner to assimilate the new knowledge and truly learn, therefore, providing the learner with the necessary learning functions in order to engage with the material provided in a meaningful way, allowing the student to correctly assimilate the concepts that are being taught.

Grosser compiled a list of those teaching functions considered of most importance based on the work conducted by Shuell and Moran. The table 4.1 contains the result of this compilation of most relevant LTFs.

4.1.2 Learning and Teaching functions for Software Engineering Education

In the previous section, the whole set of learning and teaching functions is identified. However, these functions are related to the act of teaching in general. After identifying these functions, Letra conducted a survey among professors of Software Engineering in order to achieve a subset of learning and teaching functions responsible for the best outcome of teaching in this specific field, therefore, fiding the relationship represented by **B** in the figure 4.1.

Even though the results were not unanimous, there were seven learning and teaching functions that gathered the total agreement amongst the professors:

- Attention;
- Interpreting;
- Analysing;

Table 4.1: Learning and Teaching functions

Learning and Teaching function	**Description**
Expectations	First of all, the one that is about to learn needs to have a general idea of what is the teaching he is about to be given.
Motivation	The one that is learning needs to be motivated and that motivation needs to be stimulated.
Prior knowledge activation	Students need to have in mind the required prior teachings.
Attention	Making the student focused on the lesson he is about to be taught.
Encoding	Helping the students relate what they are learning with something personal.
Comparison	Helping create higher-order relations comparing the teachings and finding something similar or different.
Hypothesis generation	Allowing the learner to create alternative solutions.
Repetition	Inducing multiple views and perspectives of the intended teaching.
Feedback	Students need to get feedback on what they are actually learning.
Evaluation	Providing the students with tools that allow them to create their own feedback and understand how well they have learnt their lesson.
Monitoring	Providing students with the tools for them to evaluate their own progress of learning.
Combination, integration, synthesis	Individual pieces of new information need to be grouped and related to facilitate the learning process.
Interpreting	Providing students with the knowledge to convert their teachings from one representation form to another.
Exemplifying	Ilustrating what is being taught with examples.
Classifying	Allow students to determine the categories in which their newly aquired concepts are inserted.
Summarizing	Providing guidance to shorten the information they learn.
Inferring	Help students reach conclusions with the given information.
Explaining	Providing cause-effect relations to explain new concepts.
Applying	Demonstrate how their newly acquired knowledge can be applied.
Analysing	Help students to divide their known concepts and to find the relation between them.
Planning	Help students think in advance of how to overcome problems.
Producing and constructing	Providing tools for the students to create new products.

- Feedback;
- Monitoring;
- Explaining;
- Applying.

In the next stage, Letra specified the relationship between these LTFs and specific GDPs or, more specifically, categories of GDPs.

4.1.3 Mapping Game Design Patterns to Learning and Teaching Functions

Having already found the relation between LTFs and the Software Engineering Education outcomes, we have a subset of LTFs that are considered to be responsible for a positive outcome.

In his work, Kelle describes a relation between the Learning and Teaching functions and Game Design Patterns [KKS11]. Using his way of relating these two concepts we get to the needed relation of the figure 4.1, the relation **a**, relating the LTFs (column 1) to their repective categories of GDPs (column 2), provided in the table 4.2.

Table 4.2: LTFs and GDPs relation

Learning and Teaching Functions	**Categories of Game Design Patterns**
Prior knowledge activation	Goals patterns
Motivation	Actions and Events patterns
Attention	Game Elements patterns
Expectation	Goals patterns and Narrative patterns
Encoding	Information patterns
Comparison	Information patterns
Repetition	Meta game patterns
Interpreting	Goals patterns
Exemplifying	Game Elements patterns
Combination, integration, synthesis	Goals patterns
Classifying	Information patterns
Summarising	Information patterns and patterns for Game Sessions
Analysing	Patterns for game mastery
Feedback	Patterns for Game mastery and Information patterns
Evaluation	Information patterns
Monitoring	Information patterns
Planning	Game Mastery patterns
Hypothesis generation	Interaction patterns
Inferring	Goal structures patterns
Explaining	Information patterns and Game Elements patterns
Applying	Game Elements patterns
Producing and constructing	Immersion patterns

4.1.4 Mapping Game Design Patterns to Software Engineering Education

Having established the previous two relations (relation **A** and relation **B** of the concept triangle in 4.1), the missing one (**c**) relates the Game Design Patterns to an expected good outcome in Software Engineering Management Education, and, therefore, needed to be validated. This was done with an experiment using the game SimSE [LPF15]. The results of the experiment validated the mapping done in the previous section, providing a set of categories of GDPs that, when present in a serious game for SEE, can ensure that the student has the necessary tools to assimilate knowledge. These GDPs can be found in the table 4.3.

Table 4.3: LTFs and GDPs relation (Letra's conclusions)

Learning and Teaching Functions	**Categories of Game Design Patterns**
Attention	Game Elements patterns
Interpreting	Goals patterns
Analysing	Patterns for game mastery
Feedback	Patterns for Game mastery and Information patterns
Monitoring	Information patterns
Explaining	Information patterns and Game Elements patterns
Applying	Game Elements patterns

4.2 Software Engineering Requirements

Another related study by Rafaela Faria [FMCS12] is a dissertation in which several Serious Games were already analysed. The author compiled the list of Game Design Patterns that can be found in the Serious Games selected for analysis.

In this study, the main goal was to achieve the Game Design Patterns most relevant to Serious Games specifically in the area of Software Requirements, having also confirmed with professionals in the area the LTFs that they consider to be more relevant in the teaching of this specific field. The method used was similar to the one previously presented and used by Pedro Letra, in his study. In this work, Faria mostly validated the resuts achieved by Letra, even if in a different area of studies, As this dissertation is about the teaching of Software Engineering, a broader approach will be taken compiling other areas of SE.

Some of the games used in the study will also be analysed in this dissertation, ensuring that the results are consistent amongst all games.

4.3 Summary

Some of the related work done in this area of studies was presented:

- Game Design Techniques for Software Engineering Management Education;
- Game Design Techniques for Software Engineering.

This work is based on both of these, as they are previous approaches to a problem that is, at least, similar to the one at hand.

In order to be able to relate the GDPs to the SEE outcome we needed to understand how the concepts of Game Design Pattern and Learning and Teaching functions relate as described in figure 4.1.

In the next chapter we can find the full survey of the Serious Games for SEE. There are two key aspects we aim to address at the end of the survey:

- Corroborate the seven LTFs identified by Pedro Letra as important for Software Enginering Education (specifically in Software Engineering Management Education) using this survey of Serious Games for Software Engineering Education;
- Expand the research in this area providing a list of the most important categories of Game Design Patterns based on the Learning and Teaching functions they are related to.

Chapter 5

Game Design Patterns for Software Engineering Education: a survey

Even though we can find a definition of Serious Games in general, this work is directed to Serious Games for Software Engineering Education. In this case specifically the serious purpose is the teaching of Software Engineering. By saying "teaching Software Engineering" it is not yet clear what kind of games should be included or not. To make this selection clear, and to ensure that the process can be repeated, the need to set a number of rules to define which games are considered Serious Games for SEE arises.

This set of rules can be defined as an Acceptance function and a Rejection function. The acceptance function has a group of conditions that, if verified, validate the Serious Game for Software Engineering Education. By the same logic, the rejection function contains another group of conditions that, if verified, marks the game as invalid to enter this group.

> "consisting in systematic observation, measurement, and experiment" [Oxf16] - description of the scientific method by the Oxford dictionary.

Considering the quote above, having a list of games without describing the method used to define how they were collected would be redundant as any scientific method needs to be described in order to be repeatable and validated by future studies. As such, the first part of the study is the definition of the method to be used when selecting the Serious Games that I am about to study.

5.1 Method description

As mentioned before, the repeatability of the method used in this survey is of most importance allowing future surveys to repeat the process corroborating the results achieved in this work. To do so, the method applied in this dissertation is an adaptation of a Systematic Survey, providing the tools for future similar surveys and future validation of the results.

> *"If studies give consistent results, systematic reviews provide evidence that the phenomenon is robust and transferable."* [Kit04]

Even though the quote above is specifically related to systematic reviews, the principle applies due to the adaptation made on our approach as shown in the figure 5.1.

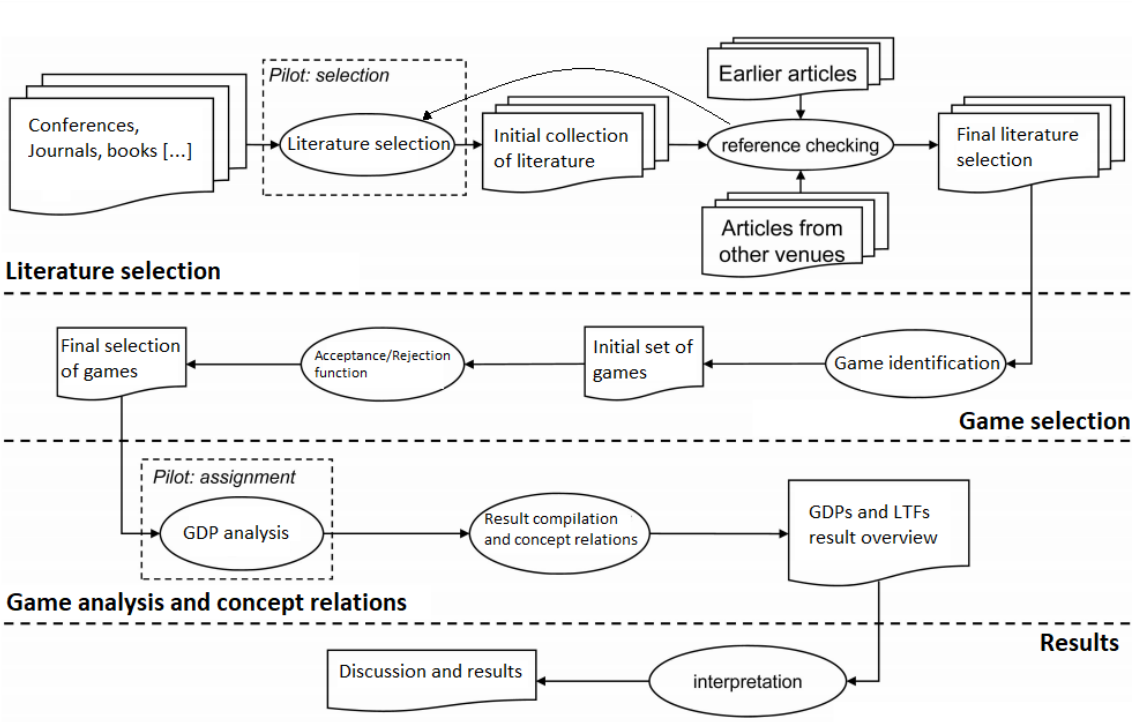

Figure 5.1: Systematic Survey adaptation (adapted from [CZvD+09])

As represented in the figure 5.1, the whole process is divided in four main parts. First of all, the selection of the literature from which the survey retrieves information was done. This part includes the selection of conferences considered for the game selection as well as cross referencing articles and other retrieved documents. Secondly, the whole process of selecting and sorting the games that were studied is described. The next part concerns the analysis of the games as well as the results it provided. Finally these results are interpreted and discussed.

5.2 Literature Selection

As described above, this is the first of four main parts of the survey. The method used in the selection of the base literature as well as the game selection is described in the following sections.

5.2.1 Literature Selection method

The selection method is based on two separate parts. First of all, the base from where every game is going to be retrieved. Saying that all games from the game industry or academic realm are going to be considered is unrealistic, as both are constantly being updated and considering all the

games would make the study of those selected to be too superficial, considering the available time to develop a masters thesis. In the next section, all the conferences and journals used to select the games that are being studied are mentioned.

This literature, from which the games are going to be selected and analysed, has already been confirmed and checked through other articles (present in those articles' reference list, for instance) being the final literature selection. In this final selection, there will be several journals regarding the same games as well as the previous work mentioned in chapter 4, which contain information on some of the games.

5.2.2 Conferences and Journals considered

As mentioned in 5.2.1, the following list contains all conferences, books or journals from which the Serious Games for this study were retrieved. For this selection, the relevance of each source in this area of studies was considered, as well as their athor's, when applicable, as is the case of the "Patterns in Game Design" book.

- XIX Simpósio Brasileiro de Informática na Educação;
- Information and Software Technologies: 20th International Conference, ICIST;
- 2015 10th Iberian Conference on Information Systems and Technologies, CISTI 2015;
- Anais do V Fórum de Educação em Engenharia de Software (FEES 2012);
- Simpósio Brasileiro de Qualidade de Software SBQS;
- 2nd International Conference on Games and Virtual Worlds for Serious Applications, VS-GAMES 2010;
- WEI 2015 - XXIII Workshop sobre Educação em Computação;
- Proceedings of the 2000 International Conference on Software Engineering. ICSE 2000 the New Millennium;
- Conferences in Research and Practice in Information Technology Series;
- Proceedings of the The 40th ACM Technical Symposium on Computer Science Education, Chattanooga, TN, March 2009;
- Proceedings of the Teaching and Learning Forum: Creating an inclusive environment: Engagement, equity and retention: Proceedings of the 21st Annual Teaching Lerning Forum;
- Empirical Software Engineering - An International Journal;
- Patterns in Game Design.

The list above represents the final literature selection, represented by the final step of the first section (**Literature selection**) of the figure 5.1.

5.3 Game Selection

5.3.1 Initial set of Games

Using the journals, books and conferences referred above, some games were identified and a static evaluation was made in order to have some information on each of them. After this evaluation these games will go through the acceptance and rejection functions to assess if they are within the conditions to follow through the study.

In this first analysis, the following fields were analysed for each of the games selected:

- Name/Title;
- Author;
- Year of creation;
- Synopsis;
- Type of Game (video-game, card game, board game, etc.);
- Reference (main source of information);
- Teaching topics (paramount for the acceptance/rejection function).

Table 5.1: SimSE - Game info

Name/Title	SimSE
Author	Emily Navarro
Created on	2010
Synopsis	This is a single-player experience where the player manages a Software development team. Apart from hiring and firing employees, based on a budget provided at the beginning of the game, the player has to manage tools and tasks according to his goals and to the information shared by his employees during gameplay. Each possible employee has different attributes and can be ordered to performed different tasks according to his own attributes. His performance depends on how well the employer allocates his employees with the tasks at hand. At the end of each game the player is given a score that represents how well he performed while developing his project.
Type of Game	Video-game
Reference	[Nav06]
Teaching Topics	• <Software Requirements; • Software Construction; • Software Testing; • Software Maintenance; • Software Engineering Models and Methods; • Software Engineering Professional Practise.

Table 5.2: SE RPG - Game info

Name/Title	SE RPG
Author	Fabiane Barreto Vavassori Benitti, Jefferson Seide Molléri
Created on	2008
Synopsis	SE RPG is a game that simulates a Software Development environment in which the player needs to interact with different characters in order to advance the development of his project. At the beginning of the game the player is introduced to a brief description of the project that he is going to develop, as well as to the budget available to the development. Based on this first piece of information, the player must choose the model of development and the language in which the project is going to be implemented. The player must also choose his development team based on the attributes of the available characters. After these first choices the player follows the development of the project assigning tasks to different characters and firing/hiring new employees. At the end of the development, the player has a choice to deliver the product to the client, at which point the game ends and a score is given to the player. A brief comment on the results also appears concerning the process, budget and time of development.
Type of Game	Video-game
Reference	[BM08]
Teaching Topics	• Software Requirements • Software Construction • Software Testing • Software Engineering Models and Methods • Software Quality

Table 5.3: PlayScrum - Game info

Name/Title	PlayScrum
Author	[a]
Created on	[a]
Synopsis	Playscrum is a board game the requires at least 2 and at maximum 5 players. Each player has a board and cards and has a different project. During the gameplay, each player is the Scrum Master of his own project. The cards that each player has define the project and the time of sprints. Following the rules of the game, the game ends when the players finish developing their projects and the winner is the one with the minimum number of errors.
Type of Game	Board/Card game
Reference	[FS10]
Teaching Topics	• Software Requirements • Software Engineering Models and Methods • Software Quality

[a] The documentation is not clear on who and when this game was created.

Table 5.4: SESAM - Game info

Name/Title	SESAM
Author	Anke Drappa and Jochen Ludewig
Created on	Simulator in SESAM created in 2000
Synopsis	SESAM is short for *Software Engineering Simulation by Animated Models*. It is the same concept as a flight simulator is for new pilots, but this time, concerning project managers. It relies on a purely textual interface, in which the player receives all the data it has to and in which it gives the orders or commands he wants. This does not mean that the player has complete information on what is happening with his project, as the simulation records and uses many different internal variables that are not available during gameplay. Through this interface the player can hire or fire staff, command new code reviews or corrections. At the end of the project the simulation ends and then the player is allowed to analyse his performance giving him a score and access to these internal variables, allowing him to draw conclusions on how well he performed as a project manager. This simulation is not aimed to teach an Engineer to become a good project manager, but to motivate him to understand how hard it can be and to make him want to actually learn what he needs in order to improve his capabilities.
Type of Game	Video game/simulation (textual interface)
Reference	[DL00]
Teaching Topics	• Software Engineering Management • Software Engineering Process • Software Quality • Software Engineering Professional Practise • Software Engineering Economics

Table 5.5: AMEISE - Game info

Name/Title	AMEISE
Author	Roland Mittermeir, Elke Hochmüller, Andreas Bollin, Susanne Jäger and Markus Nusser
Created on	Extension of SESAM created in 2001
Synopsis	AMEISE is an extension built on SESAM simulations. It has a different interface and allows players to make a SESAM simulation a competitive experience, allowing them to compare results and repeat a part of their simulation without having to do it all over again. Considering AMEISE as a teaching tool it removes from the instructors hand the task of evaluating all of their student's performance, evaluating part of it and providing their results.
Type of Game	Video game/simulation
Reference	[MHB+]
Teaching Topics	• Software Engineering Management • Software Engineering Process • Software Quality • Software Engineering Professional Practise • Software Engineering Economics

Table 5.6: SimJavaSP - Game info

Name/Title	SimJavaSP
Author	Katherine Shaw and Julian Dermoudy
Created on	2005
Synopsis	SimJavaSP is another simulation of a Software development team. The player is the team manager and is responsible for hiring, firing and managing the tasks of his staff, as well as for being aware of the budget allowed. One of the functionalities of this game is that it tries to apply random events to the equation, as they are quite common in the real world which is the goal of these Serious Games. The game ends when the project is 100% complete, at which point the player gets his result for the project's quality, time spent and remaining budget.
Type of Game	Video game
Reference	[SD05]
Teaching Topics	• Software Requirements • Software Configuration Management • Software Engineering Management • Software Engineering Process • Software Engineering Models and Methods • Software Quality • Software Engineering Professional Practise • Software Engineering Economics

Table 5.7: iTest Learning - Game info

Name/Title	iTest Learning
Author	Virgínia Farias, Carla Moreira, Emanuel Coutinho and Ismayle S. Santos
Created on	2012
Synopsis	iTest Learning is a single-player game where the player has to develop a plan of how to test a specific project based on it's specification provided at the beginning of the game. The player is allowed to access concept definitions and explanations during gameplay and to access the project's specification during any part of the test planning. At the end, the player is shown his chosen test planning and the optimal one so that he can compare both and understand his mistakes.
Type of Game	Video game
Reference	[FMCS12]
Teaching Topics	• Software Testing

Table 5.8: XMED - Game info

Name/Title	XMED
Author	Lino JI
Created on	2007
Synopsis	XMED is a game that allows the player to follow the flow of a project since it's beginning until it's delivery. During the projects development several choices are given to the player. For instance, after a brainstorming meeting, a textual transcript is given to the player. As in every other choice, 6 options are given. After choosing one the game goes on, however, it goes on using the correct choice regardless of the player's choice, giving him feedback for each choice he makes. At the end of the game, a complete feedback is given to the player showing him his mistakes and bad/good choices in order for him to improve in future runs.
Type of Game	Video game
Reference	[GTK09]
Teaching Topics	• Software Requirements • Software Maintenance • Software Configuration Management • Software Engineering Management • Software Engineering Process • Software Engineering Models and Methods • Software Quality • Software Engineering Professional Practise • Software Engineering Economics

Table 5.9: A Ilha dos Requisitos - Game info

Name/Title	A Ilha dos Requisitos
Author	Marcello Thiry, Alessandra Zoucas and Rafael Queiroz Gonçalves
Created on	2010
Synopsis	A Ilha dos Requisitos is a game foccused on teaching the importance of Software Requirements. The environment it presents is an unknown island in which "Jack", the main character, crashes. Nothing about the story surrounding him has anything to do with Software Development, however, he is charged with defining who in the island does what in order to flee the island before the volcano erupts, trying to mimic a situation in which the player has to define the Software Requirements without mentioning them. This game differs from many other with the same goal precisely because of it's attempt to teach this topic without directly relating to it.
Type of Game	Video game (adventure/strategy)
Reference	[TZG10]
Teaching Topics	• Software Requirements

Table 5.10: SimSoft - Game info

Name/Title	SimSoft
Author	Jianhong (Cecilia) Xia, Craig Caulfield, David Baccarini and Shelley Yeo
Created on	2012
Synopsis	SimSoft is a Serious Game aimed at teaching assessing Risk Management to players. The game starts with a small survey in order to understand how proficient the player is in this area. After this first phase, the project's description and details are provided to him. After the player's study of the project, the game goes through every development phase of the project. During each phase, the player is asked a series of short/multiple choice questions, after which immediate feedback is provided in order the allow the player to understand whether he is correct or incorrect and why he is so. Depending on his answer, the player might loose money or gain score points. When the game ends, the player is shown his score and remaining money and feedback. At the end, another survey is made in order to asses whether the player learned anything or not.
Type of Game	Video game
Reference	[XCBY12]
Teaching Topics	• Software Maintenance • Software Engineering Models and Methods • Engineering Foundations

Table 5.11: O Jogo das 7 Falhas - Game info

Name/Title	O Jogo das 7 Falhas
Author	Lucio L. Diniz and Rudimar L. S. Dazzi
Created on	2002
Synopsis	*"O Jogo das 7 Falhas"* is a single-player game in which the goal is for the player to find the 7 errors in a simple program similar to any "register" form. The player can test the program as he likes but has to find these errors in 25 minutes. After finding an error, the player has to answer what might be the origin of this error. This game has also another level, in which the program with the errors is a bit more complex, however, the mechanics are the same.
Type of Game	Video game
Reference	[DDAU11]
Teaching Topics	• Software Design • Software Construction • Software Testing

Table 5.12: iLearn Test - Game info

Name/Title	iLearn Test
Author	Tânia P. B. Ribeiro and Ana C. R. Paiva
Created on	2015
Synopsis	iLearn Test is a single-player game aimed at teaching Software Testing. It's menu works as a platform game in which each platform represents a different lesson/mini game. Each game has a different concept. Some work as the hangman game, some as fill the blanks with the words, etc.. Some lessons teach the difference between black and white box testing, some of them teach test design techniques and many other areas trying to accommodate all areas of software testing. For each lesson, a score is awarded. In the platform menu, the maximum score for each lesson is accessible by the player.
Type of Game	Video game
Reference	[RP15]
Teaching Topics	• Software Design • Software Testing

Table 5.13: CRobots - Game info

Name/Title	CRobots
Author	Tom Poindexter
Created on	1985
Synopsis	CRobots is a game that works best if played by several players. In most games, the player has an impact during gameplay, however, in CRobots, the player only has an impact before gameplay. Each player has a different robot. All the player's robots are equally equipped, however, their behaviour is defined by the player's C program. Basically, the player programs the robot to seek and destroy the other players' robots. The robot that survives becomes the winner.
Type of Game	Video game
Reference	[cro]
Teaching Topics	• Software Construction • Software Configuration Management

5.3.2 Acceptance function

As shown in the figure 5.1, the next part of this study is to get the first group of games selected and make them go through the Acceptance and Rejection function. Both these functions contain a list of conditions. In order to be considered to the study the game must fulfil the conditions of the Acceptance function and must not fulfil any condition of the rejection function. After this, the remaining games are the ones considered in the following steps.

As mentioned in 3.2, the acceptance function is the group of conditions that a game has to fulfil in order to be considered to this study. In this case, to consider a game it needs to:

- Be a Serious Game (consistent with the definition provided in the chapter 3);
- Be mentioned in at least one of the sources mentioned in the subsection 5.2.2;
- Have as its teaching topics at least one of KA of SE different from Computing Foundations, Mathematical Foundations and Engineering Foundations;
- Be a playable game and not only a concept (Even if the runnable version of the game can not be found at this time, the game needs to have been playable).

By definition, a Rejection function is the set of conditions from which, if a game fulfils at least one, it is rejected. In this case, the rejection function does not exist as it would be the exact opposite of this one which means that it is redundant to define it.

As previously described, making the games go through this acceptance function ends the process of selecting the games for the study. The results are presented in the next section of this document.

5.3.3 Final Selection of Games

After the whole selection method, the final set of games to be analysed for this study (containing thirteen games) is as follows: SimSE, SE RPG, PlayScrum, SESAM, AMEISE, SimJavaSP, iTest Learning, XMED, A Ilha dos Requisitos, SimSoft, O Jogo das 7 Falhas, iLearn Test, CRobots.

Defining an Acceptance function for the game selection is an important part of the process as it ensures the validity of the games and the repeatability of this study. After the first selection of games, each game was analysed and found to fulfill the requirements set by this Acceptance function, being selected to further study.

5.4 Game analysis and concept relations

In this section we can find a description of the method used to analyse the selected games as well as the results achieved with this analysis. Both of these are represented as the **Game analysis and concept relations** section of the figure 5.1. Note that not all of the results will be found in this chapter as they are quite extensive. The results of the complete analysis can be found in the Appendix B.

5.4.1 Game Design Pattern analysis' method

In order to make the game analysis as consistent as possible, a method needs to be defined so that all games are analysed as equally as possible. In the section 3.3.2 of the third chapter, we can find two different types of analysis defined by Bjork.

As described before, these types of analysis require the existence of certain items. For instance, in order to be able to Play Test each game, a runnable version of each game is required. As it happens, some of the selected games do not have, at the moment of the study, an available runnable version, which would make this analysis by itself impossible to rely on. Regarding the available documents for each game, even though many information can be found, not all the same documents are found for every game. For instance, an instruction manual can be found for some of them but not for all.

In order to achieve results as complete and accurate as possible, the analysis performed in this study attempted to compile all the possible ways of analysing a game's GDPs ensuring that the results are correct based upon the available material. First, a structural like analysis was performed for each game, using all the available literature found on that game. After all the possible information was retrieved from these sources, a runnable version was played (if existent for the game in question). Note that this analysis was not exactly the same as Play Testing, as it requires someone that is not aware of the analysis to play. While playing, the information retrieved from the structural analysis was confirmed and even completed, as some patterns could be better identified while playing.

As an example of this analysis, a screenshot and brief explanation of why does the GDP **Penalties** of the **Actions and Events** category exists in the game *A ilha dos Requisitos* is shown below.

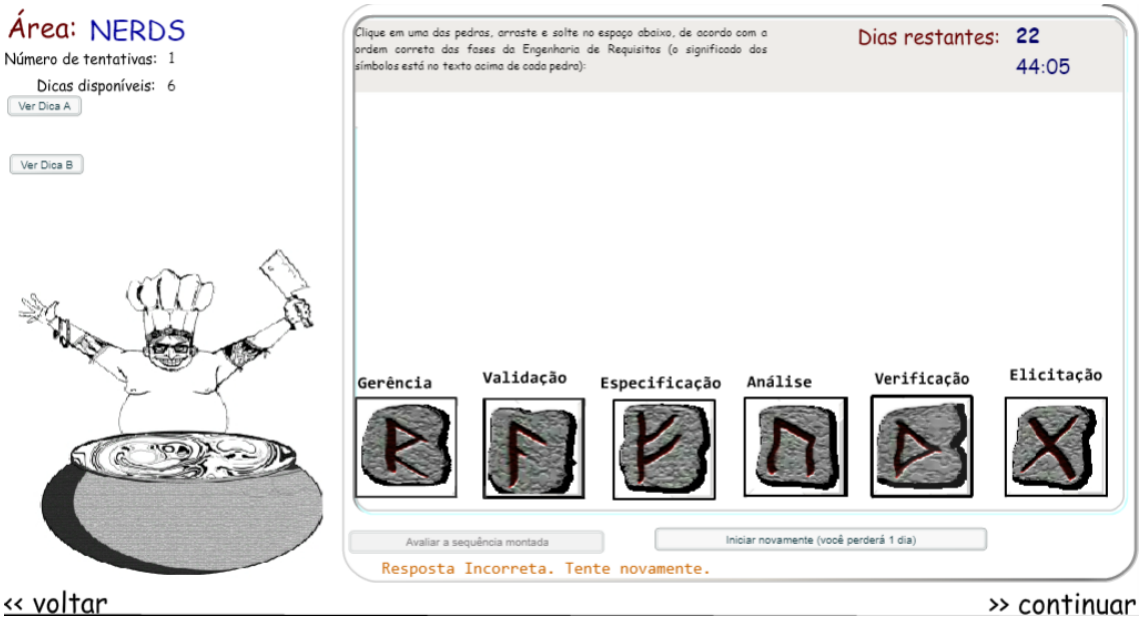

Figure 5.2: A ilha dos Requisitos - gameplay screenshot

From a brief analysis of the screenshot provided in 5.2, some patterns can be recognized besides the one being analysed at the moment. However, concerning the pattern **Penalties**, the game informs the player that he failed to answer correctly and demands one day of the player's time (resource) to try again, penalising the player for not answering correctly to the question. In this case, even the brief core description of the pattern is enough to assert that this pattern is present:

> "Players are inflicted with something perceived as negative or stripped of an advantage, due to failure to meet a requirement in the game." [BH05]

Even though, in this case, the pattern in pretty straightforward, most of the times a detailed analysis of the complete description and "using the pattern" fields is required to understand if the pattern is present or not.

5.4.2 Game Design Pattern analysis' results

The complete GDPs analysis was made, identifying the existence, or not, of each of the GDPs in each of the games. The complete analysis' result can be found on the table B.1 of the Appendix B. In that table, a raw compilation of the data collected can be found, however, the format in which the information is presented is not ideal to draw any sort of conclusions.

First of all, we tried to understand the extent to which each of the category of Game Design Patterns is used. To do so, a relationship between the number of GDPs of each category and the number of GDPs found of that specific category was made as shown in the table 5.14.

Table 5.14: GDPs appearance ratio

GDP group	GDPs/group	Total	Ratio Total/(GDPs/Group)
Game Elements	26	79	0.33
Resource and Resource Management	6	18	0.33
Information, Communication and Presentation	8	32	0.25
Actions and Events	24	72	0.33
Narrative Structures, Predictability and Immersion	19	54	0.35
Social Interaction	16	10	1.60
Goals	16	25	0.64
Goal Structures	14	28	0.50
Game Sessions	11	44	0.25
Game Mastery and Balancing	22	80	0.28
Meta Games, Replayability and Learning Curves	4	9	0.44

In order to better understand the significance of these results, the GDPs' appearance ratio was represented in the figure 5.3. As we can understand from the graph, there is a tendency for the representation of the categories, represented by their ratio, to be close to 0.3281, however,

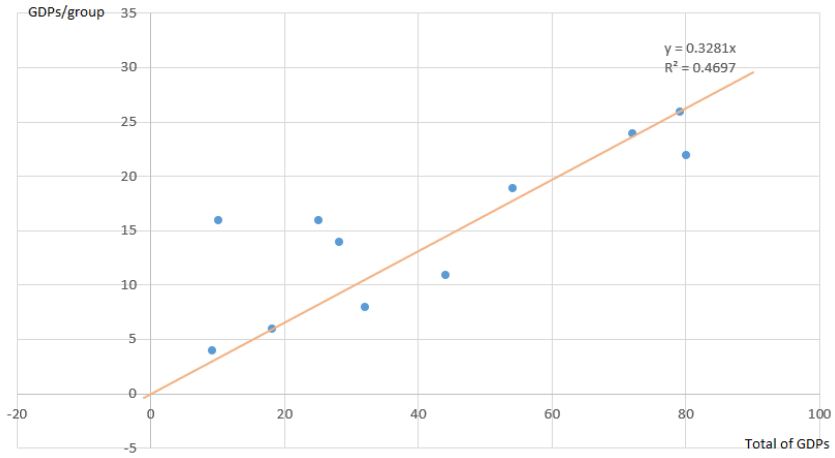

Figure 5.3: Game Design Patterns appearance ratio

some of the categories' ratios have values significantly different from that value. The **Social Interaction** category is, by far, under-represented in these games. That might be a designer's choice, considering that very few of the games selected support any kind of multi-player.

The next phase of this analysis was to understand which categories of GDPs are represented in each game. The first approach was to consider that for a category to be represented, the game needed to include at least one of the GDPs of that category, however, this is not very relevant or accurate, as some GDPs are too common and some categories contain twice as many GDPs as others. To analyse the categories' representation in a representative way, we compiled the percentage of GDPs of each category present in each of the games (result in the table 5.15).

Table 5.15: Percentage of GDPs per Game

GDP group/Number of patterns	**Sim SE**	**SE RPG**	**PlayScrum**	**SESAM**	**AMEISE**	**SimJavaSP**	**iTest Learning**	**XMED**	**A Ilha dos Requisitos**	**SimSoft**	**O Jogo das 7 Falhas**	**iLearn Test**	**CRobots**
Game Elements	**30.8%**	26.9%	3.8%	15.4%	23.1%	26.9%	19.2%	15.4%	**46.2%**	19.2%	19.2%	**30.8%**	26.9%
Resource and Resource Management	**50.0%**	**33.3%**	**66.7%**	**33.3%**	**33.3%**	**50.0%**	0.0%	0.0%	**33.3%**	0.0%	0.0%	0.0%	0.0%
Information, Communication and Presentation	25.0%	25.0%	**50.0%**	25.0%	25.0%	**37.5%**	25.0%	25.0%	25.0%	25.0%	25.0%	25.0%	**62.5%**
Actions and Events	25.0%	25.0%	**33.3%**	20.8%	25.0%	29.2%	16.7%	16.7%	25.0%	16.7%	16.7%	29.2%	20.8%
Narrative Structures, Predictability and Immersion	26.3%	21.1%	26.3%	26.3%	26.3%	26.3%	10.5%	10.5%	**42.1%**	10.5%	21.1%	15.8%	21.1%
Social Interaction	0.0%	0.0%	25.0%	0.0%	12.5%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	25.0%
Goals	12.5%	6.3%	**31.3%**	12.5%	12.5%	6.3%	0.0%	6.3%	18.8%	0.0%	25.0%	12.5%	12.5%
Goal Structures	14.3%	7.1%	**35.7%**	14.3%	21.4%	14.3%	7.1%	7.1%	7.1%	7.1%	14.3%	14.3%	**35.7%**
Game Sessions	27.3%	27.3%	27.3%	27.3%	27.3%	27.3%	27.3%	27.3%	**36.4%**	27.3%	**36.4%**	**36.4%**	**45.5%**
Game Mastery and Balancing	27.3%	27.3%	27.3%	**31.8%**	**31.8%**	**31.8%**	22.7%	18.2%	18.2%	27.3%	**45.5%**	**31.8%**	22.7%
Meta Games, Replayability and Learning Curves	25.0%	25.0%	25.0%	25.0%	25.0%	25.0%	0.0%	0.0%	0.0%	0.0%	**50.0%**	25.0%	0.0%

Having the percentage of representation of each category of GDPs in each game, allows us to complete the relation between Catgories of GDPs and LTFs, providing us with the representation of each Learning and Teaching function in each game, as shown in the table 5.16. The existence of a LTF in a game is considered when the respective categories of GDPs are represented above 30.0%.

Table 5.16: Learning and Teaching Functions per Game

Learning function	Sim SE	SE RPG	PlayScrum	SESAM	AMEISE	SimJavaSP	iTest Learning	XMED	A Ilha dos Requisitos	SimSoft	O Jogo das 7 Falhas	iLearn Test	CRobots	Total
Prior knowledge activation			X											1
Motivation			X											1
Attention	X								X			X		3
Expectation			X						X				X	3
Encoding			X			X							X	3
Comparison			X			X							X	3
Repetition											X			1
Interpreting			X											1
Exemplifying	X								X			X		3
Combination, integration, synthesis			X											1
Classifying			X			X							X	3
Summarising			X			X			X		X	X	X	6
Analysis				X	X	X					X	X		5
Feedback			X	X	X	X					X	X	X	7
Evaluation			X			X							X	3
Monitoring			X			X							X	3
Planning				X	X	X					X	X		5
Hypothesis Generation			X											1
Inferring			X										X	2
Explaining	X		X			X			X			X	X	6
Applying	X								X			X		3
Producing and Constructing									X					1

After this relation was established, we needed to make sense of this information, trying to understand what Learning and teaching Functions are more represented and relevant to the Serious Games for SEE. To do so, the representation of each LTF was ordered and represented in the figure 5.4.

5.5 Discussion and results' comparison

In this section we will start by comparing the achieved results with the results from previous studies. This will allow us to validate or refute previous results, as well as validating the results of this study. Keeping in mind that the main two previous works considered are specified within SEE (to Software Engineering Management Education and to Software Requirements Education), the comparison between these results is of most importance as these are two specific areas of Software Engineering.

After the complete analysis of the games and complete mapping of the Learning and Teaching functions from the Game Design Patterns, a set of these LTFs is shown in the figure 5.4, ordered from most present (considered to be most relevant) to least present. Letra also identified a set of the seven most important Learning and Teaching functions for the teaching of Software Engineering management. In the figure 5.5 we can find our results, highlighting Letra's seven LTFs in orange.

The figure 5.5 shows that there is one of the Learning and Teaching functions identified by Letra (filled columns) that is less represented than most in our analysis: Interpreting. However, considering the rest of the LTFs, six of the seven LTFs considered the most important by this previous work are in the top eight LTFs most present in our study. One of the main reasons for this to happen is the fact the the set of games used to analyse the GDPs did not just include

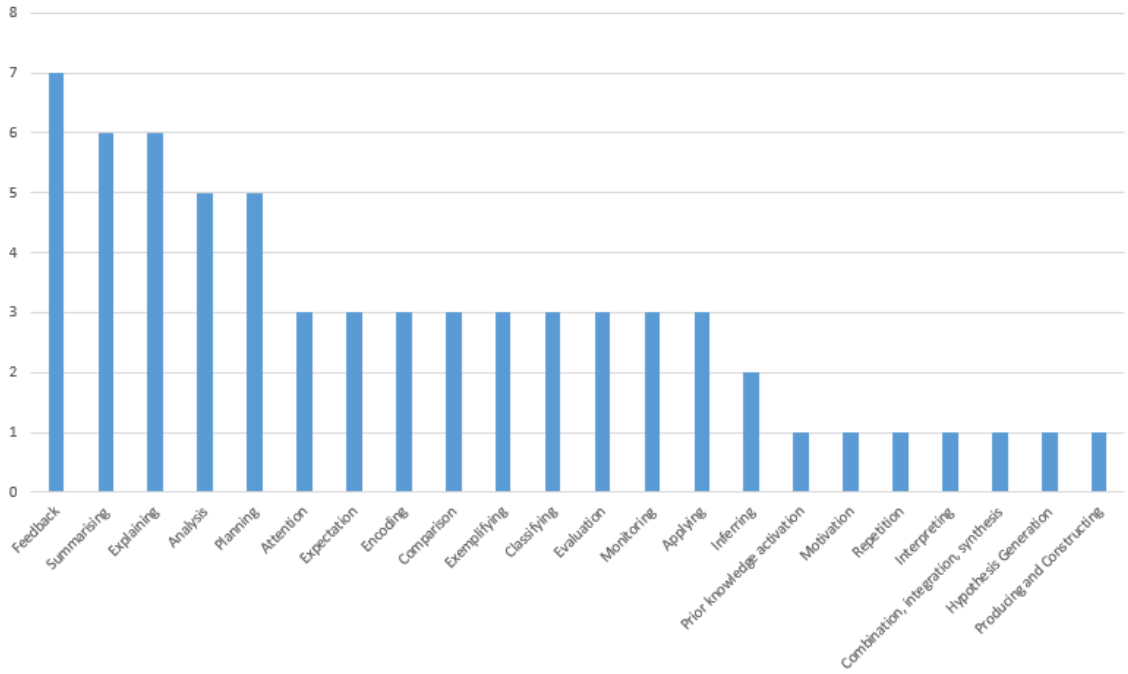

Figure 5.4: Representation of each Learning and Teaching Function

Software Engineering Management, as this work concerns a broader topic, Software Engineering as a whole.

From our results, we can also notice that, apart from these already studied LTFs, Summarising and Planning are two LTFs that are present in most of the Serious Games for SEE, strongly suggesting that they might be considered as important to the good outcome of teaching as the other LTFs.

5.6 Threats to validity

Even considering that the results are quite satisfactory and based on reliable sources, it is important to note that there are threats to their validity that can be minimized in the future. Three major threats are described below:

- Subjectivity of analysis
  - **Threat** - As the analysis of the GDPs within each game is done based on the description of a pattern, different people might not agree on the existence or absence of a GDP within a game;
  - **Used solution** - In this work the analysis was conducted by one person ensuring the consistency of the analysis' results;
  - **Optimal solution** - In order to minimize the subjectivity of the results, multiple people are needed to analyse each game, discussing or reaching a consensus on the drawn

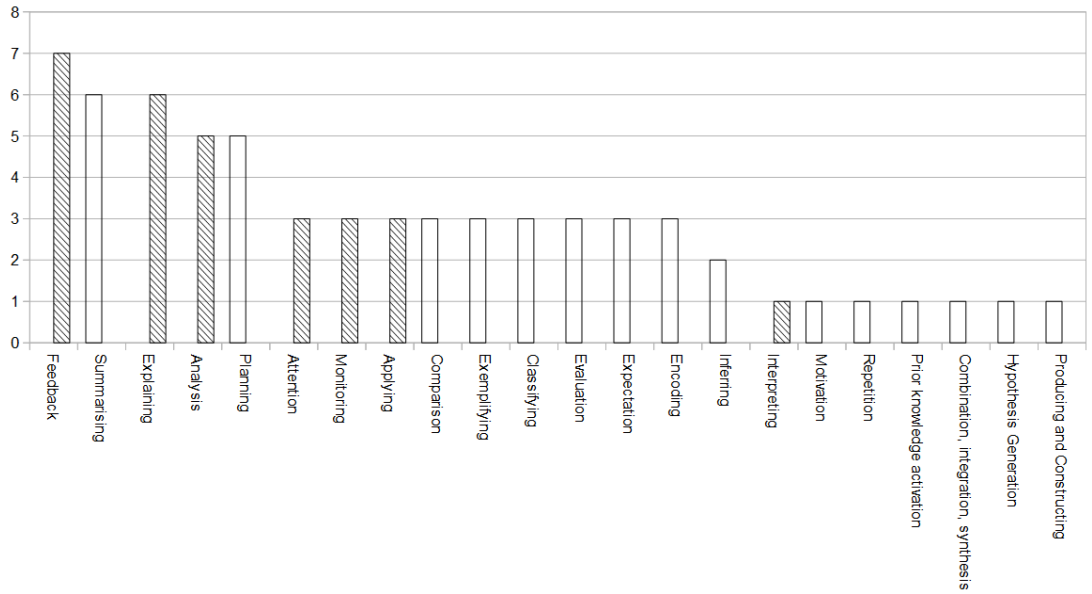

Figure 5.5: Representation of each Learning and Teaching Function (comparison)

conclusion. This group should be the same for each game and GDP ensuring the consistency of these results;

- Existence of bias
  - **Threat** - The possibility of affecting the results due to knowing the goals of the survey;
  - **Used solution** - This analysis was carried by one analyser. Being aware of the possible existence of bias, a special effort to remain neutral throughout the analysis existed, attempting to minimize the consequences of this threat;
  - **Optimal solution** - Ideally the analysis would be conducted by people unaware of the analysis' goals and unaware of each other, providing completely bias free results;
- Diverse sources of information
  - **Threat** - Using different sources of information can compromise the consistency of the analysis' results;
  - **Used solution** - Considering that not all games had the same type of source available, this analysis was based on all available material maximizing the complexity of the results and minimizing the need to assume the existence or absence of patterns in any of the games;
  - **Optimal solution** - Ideally the analysis would be made with all the sources for all games, ensuring the maximization of sources of information and minimizing the difference of sources amongst the analysed games.

5.7 Summary

At the end of this chapter a repeatable selection method has already been described and applied. A set of thirteen games was selected and analysed, having described each step of the process ensuring it's repeatability in the future.

In the future, if this work is repeated, the number of games might be bigger as there might be new publications and other Serious Games.

Having collected and related all the information concerning which GDPs are represented in each game, and established the relation between these GDPs and Learning and Teaching functions, we have already drawn some conclusions about the presence of certain categories of GDPs in the games and compared the results to previous work in the area, as shown in the figure 5.5. The discussion of the results can be found in the last section of this chapter 5.5, not only validating Letra's conclusions concerning SEE but also identifying other significantly represented LTFs suggesting the need to further study on their relevance for SEE.

Some threats to validity were also identified. Identifying these threats allows the existence of suggestions that minimize these threats in future work, providing grounds for improvement in the research of this field.

Chapter 6

Conclusions and Future work

This chapter is related to the contribution made by this survey to the community and to the future work that might be done in the area of studies.

6.1 Contribution

In the last chapter, the results of this survey and the conclusions of previous works concerning Learning and teaching Functions responsible to a good outcome in Software Engineering Education were compared and discussed. Evidence was collected that indicates what LTFs are of most relevance. This evidence also indicates that the conclusions of this work mostly coincide with the previous work's. However, there was another analysis, from which other conclusions can be drawn, that were not yet analysed.

The figure 5.3, as said in chapter 5, suggests that not all categories of GDPs are equally represented. One of the categories that definitely is further away from the others, is the category of **Social Interaction**. There is not enough information to say that this GDP is more or less relevant to the teaching of SEE than the others, however, it is known that the low representation of this category is due to the fact that a very low number of the analysed games supports any kind of multi-player. Having multi-player would not only make this category much more represented, as it might also increase the entertainment and engagement facet of the games, contributing to the balance we spoke of in the beginning of this work between teaching and the player's engagement, as no Serious Game for Education can fulfil it's goal if the players is not engaged.

Concerning the most important Game Design Patterns, evidence suggests, using the mapping of GDPs and LTFs explained in the sub section 4.1.3, that the most relevant GDPs for SEE (top five) are as follows:

- Feedback;
- Summarising;

- Explaining;
- Analysis;
- Planing.

Converting these LTFs to GDPs using the previously mentioned mapping in table 4.2, it is possible to reach the conclusion that the top three of Game Design Patterns, expectedly more relevant for the teaching of SEE are:

- Patterns for Game Mastery;
- Information patterns;
- Patterns for Game Sessions.

From this study we hope to have enhanced the way future developers face the challenge of developing a game when trying to teach SE. Having identified the most relevant GDPs we believe to have provided future developers with new and valuable information in order for them to develop the most efficient and engaging Serious Game for Software Engineering Education as possible.

6.2 Future work

Even though the results of this dissertation are satisfying, and the conclusions drawn from previous works were validated, some improvements are possible in future approaches to this problem.

First of all, the sources of the games can be broader. This can result in a bigger set of games, as well as a more varied source of information. One other way to make this analysis even more accurate is to make sure that the analysed games have runnable versions. In this case, if that requirement existed, only a very few number of games would be possible to analyse, possibly making this analysis inconclusive.

As an improvement of this work, replaying this survey would not only be able to add credibility to these results, but also to tackle the threats to validity mentioned in the section 5.6, adopting the optimal solutions to those identified threats.

With future work on this area, even if it is in a specific area of SE, as some of the previous works are, it is expected to see an improvement in Serious Games for SEE, not only in quality and number, but also on their usage, as most of the students of SE do not use, at the moment, any of these games in order to learn what they are studying.

References

[Abt87] C.C. Abt. *Serious Games*. University Press of America, 1987.

[BAB+06] Stefan Biffl, Aybüke Aurum, Barry Boehm, Hakan Erdogmus, and Paul Grünbacher. *Value-based software engineering*. Springer-Verlag Berlin Heidelberg, 2006.

[BF14] Pierre Bourque and Richard E. Fairley. *Guide to the Software Engineering - Body of Knowledge.* IEEE Computer Society Press, 2014.

[BH05] Staffan Björk and Jussi Holopainen. *Patterns in Game Design*, volume 54. Jenifer Niles, 2005.

[BM08] FBV Benitti and JS Molléri. Utilização de um RPG no ensino de gerenciamento e processo de desenvolvimento de software. *WEI-Workshop sobre Educação em ...*, (September 2015):258–267, 2008.

[BÖBH16] Matthias Budde, Rikard Öxler, Michael Beigl, and Jussi Holopainen. Sensified Gaming: Design Patterns and Game Design Elements for Gameful Environmental Sensing. *Proceedings of the 13th International Conference on Advances in Computer Entertainment Technology*, pages 3:1—-3:8, 2016.

[cro] CROBOTS. http://corewar.co.uk/crobots.htm. Accessed: 2018-07-29.

[CZvD+09] B. Cornelissen, A. Zaidman, A. van Deursen, L. Moonen, and R. Koschke. A systematic survey of program comprehension through dynamic analysis. *IEEE Transactions on Software Engineering*, 35(5):684–702, Sept 2009.

[DAJ11] Damien Djaouti, Julian Alvarez, and Jean-Pierre Jessel. Classifying serious games: The G/P/S model. *Handbook of research on improving learning and motivation through educational games: Multidisciplinary approaches*, (2005):118–136, 2011.

[DDAU11] Lucio L Diniz, Rudimar L S Dazzi, Computação Aplicada, and Itajaí Univali. Jogo para o Apoio ao Ensino do Teste de Caixa-Preta. In *SBIE - Simpósio Brasileiro de Informática na Educação*, pages 426–435, 2011.

[DL00] a. Drappa and J. Ludewig. Simulation in software engineering training. *Proceedings of the 2000 International Conference on Software Engineering. ICSE 2000 the New Millennium*, pages 199–208, 2000.

[dS17] S. C. dos Santos. Pbl-see: An authentic assessment model for pbl-based software engineering education. *IEEE Transactions on Education*, 60(2):120–126, May 2017.

[Far16] Rafaela Faria. Game Design Techniques for Software Engineering. 2016.

[FMCS12] Virgínia Farias, Carla Moreira, Emanuel Coutinho, and Ismayle S Santos. iTest Learning : Um Jogo para o Ensino do Planejamento de Testes de Software. *Anais do V Fórum de Educação em Engenharia de Software (FEES 2012)*, pages 01–08, 2012.

[FS10] João M. Fernandes and Sónia M. Sousa. PlayScrum - A card game to learn the scrum agile method. In *2nd International Conference on Games and Virtual Worlds for Serious Applications, VS-GAMES 2010*, pages 52–59, 2010.

[Grö07] Mary Grösser. Effective teaching : linking teaching to learning functions. *South African Journal of Education*, 27(1):37–52, 2007.

[GTK09] Christiane Gresse Von Wangenheim, Marcello Thiry, and Djone Kochanski. Empirical evaluation of an educational game on software measurement. *Empirical Software Engineering*, 14(4):418–452, 2009.

[Imp] Programme Specification (Undergraduate). Technical report.

[Kir02] John Kirriemuir. Video gaming, education and digital learning technologies. *D-lib Magazine*, 8(2):7, 2002.

[Kit04] Barbara Kitchenham. Procedures for performing systematic reviews. *Keele, UK, Keele University*, 2004.

[KKS11] Sebastian Kelle, Roland Klemke, and Marcus Specht. Design patterns for learning games. 3, 01 2011.

[LPF15] Pedro Letra, Ana C. R. Paiva, and Nunes Flores. Game Design Techniques for Software Engineering Management Education. *18th International Conference on Computational Science and Engineering*, pages 192–199, 2015.

[Man] Software Engineering 2 - course unit details - BSc Software Engineering - course details (2018 entry) | The University of Manchester.

[MHB$^+$] R T Mittermeir, E Hochmüller, A Bollin, S Jäger, M Nusser, and Universität Klagenfurt. AMEISE – A Media Education Initiative for Software Engineering Concepts, the Environment and Initial Experiences. pages 1–17.

[MSS04] Alice Mitchell and Carol Savill-Smith. *The use of computer and video games for learning*. 2004.

[Nav06] E Navarro. SimSE: a software engineering simulation environment for software process education. *Vasa*, page 321, 2006.

[Oh02] E Oh. Teaching software engineering through simulation. . . . *International Conference on Software Engineering* . . . , 2002.

[Oxf16] Oxford English Dictionary. Oxford English Dictionary Online, 2016.

[Pre01] Marc Prensky. Fun , Play and Games : What Makes Games Engaging. *Digital Game-Based Learning*, 2001.

[PSM13] Pantelis M. Papadopoulos, Ioannis G. Stamelos, and Andreas Meiszner. Enhancing software engineering education through open source projects: Four years of students' perspectives. *Education and Information Technologies*, 18(2):381–397, 2013.

REFERENCES

[RP15] T P B Ribeiro and A C R Paiva. ILearnTest: Educational game for learning software testing [ILearnTest: Jogo Educativo para Aprendizagem de Teste de Software]. In *2015 10th Iberian Conference on Information Systems and Technologies, CISTI 2015*, 2015.

[Saw03] Ben Sawyer. Serious Games: Improving Public Policy through Game-based Learning and Simulation. Technical report, 2003.

[SD05] Katherine Shaw and Julian Dermoudy. Engendering an Empathy for Software Engineering. In *Conferences in Research and Practice in Information Technology Series*, volume 42, pages 135–144, 2005.

[SZ04] Katie Salen and Eric Zimmerman. Rules of Play: Game Design Fundamentals. *Nihon Ronen Igakkai zasshi. Japanese journal of geriatrics*, page 672, 2004.

[Tch11] Pierre Tchounikine. Computer Science and Educational Software Design. *Media*, pages 31–56, 2011.

[TZG10] Marcello Thiry, Alessandra Zoucas, and Rafael Queiroz Gonçalves. Promovendo a Aprendizagem de Engenharia de Requisitos de Software Através de um Jogo Educativo. *XIX Simpósio Brasileiro de Informática na Educação*, 2010.

[XCBY12] J (C.) Xia, C Caulfield, D Baccarini, and S Yeo. Simsoft: A game for teaching project risk management. In *Proceedings of the Teaching and Learning Forum: Creating an inclusive environment: Engagement, equity and retention: Proceedings of the 21st Annual Teaching Lerning Forum*, 2012.

REFERENCES

Appendix A

Game Design Patterns

Table A.1: Game Design Patterns [BH05]

Categories of GDPs	Subcategories of GDPs	GDPs
Game Elements	Game Worlds	Game World
		Reconfigurable Game World
		Levels
		Inaccessible Areas
		Consistent Reality Logic
		Alternative Reality
		Moveable Tiles
	Objects	Enemies
		Boss Monsters
		Deadly Traps
		Obstacles
		Avatars
		Units
		Tools
		Controllers
		Alarms
		Pick-Ups
		Power-Ups
		Clues
		Extra-Game Information
	Abstract Objects	Score
		High Score Lists
		Lives
	Locations	Strategic Locations
		Outstanding Features
		Chargers
Resource and Resource Management	Types od Resources	Resources
	Resource Control	Producer-Consumer
		Ownership
		Resource Management
	Resource Control	Investments
		Diminishing Returns
Information, Communication and Presentation	Information Quality	Imperfect Information
		Perfect Information
		Uncertainty of Information
	Information Distribution	Symmetric Information
		Asymmetric Information

		Public Information
	Information Access	Communication Channels
	Information Presentation	Game State Overview
Actions and Events	Actions	Combat
		Movement
		Maneuvering
		Aim & Shoot
		Construction
	Action Control	Priviledged Abilities
		Asymmetric Abilities
		Limited Set of Actions
		Downtime
		Experimenting
		Trasnfer of Control
		Interruptible Actions
		Focus Loci
		New Abilities
		Improved Abilities
		Ability Losses
		Decreased Abilities
		Extended Actions
		Irreversible Actions
		Save-Load Cycles
	Rewards and Penalties	Rewards
		Penalties
		Illusionary Rewards
	Events	Ultra-Powerful Events
Narrative Structures, Predictability and Immersion	Evaluation	Delayed Effects
		Hovering Closures
		Illusion of Influence
		Perceived Chance to Succeed
	Immersion	Immersion
		Anticipation
	Creative Control	Freedom of Choice
		Creative Control
		Storytelling
	Narrative Structires	Narrative Structures
		Tension
		Characters
		Character Development
		Planned Character Development
		Identification
		Higher-level Closure as Gameplay Progresses
		Surprises
		Cut Scenes
		Easter Eggs
Social Interaction	Competition	Competition
		Conflict
		Player Killing
		Betrayal
	Collaboration	Cooperation
	Group Activities	Team Play
		Alliances
		Roleplaying
		Contructive Play

		Player Decided Results
	Stimulated Social Interaction	Social Interaction
		Trading
		Bidding
		Bluffing
		Negotiation
		Social Dilemmas
Goals	Goals of Ownership and Overcoming Opposition	Gain Ownership
		Overcome
		Stealth
		Eliminate
		Rescue
		Capture
		Evade
		Conceal
		Race
	Goals of Arrangement	Collection
	Goals of Persistence	Guard
		Survive
		Traverse
	Goals of Information and Knowledge	Gain Information
		Gain Competence
		Exploration
Goal Structures	Goal Characteristics	Predefined Goals
		Dynamic Goal Characteristics
		Optional Goals
		Interferable Goals
		Player Defined Goals
	Relations between Goals	Preventing Goals
		Hierarchy of Goals
		Tournaments
		Incompatible Goals
		Selectable Sets of Goals
		Supporting Goals
	Relations between Goals and Players	Symmetric Goals
		Asymmetric Goals
		Commited Goals
Game Sessions	Game and Play Sessions	Real-Time Games
		Asynchronous Games
		Syncronous Games
		Single-Player Games
		Multiplayer Games
		Turn-based Games
		Closure Points
	Player Activity	Player Elimination
		Analysis Paralysis
		The Show Must Go On
		Agents
Game Mastery and Balancing	Game Mastery	Game Mastery
		Empowerment
		Timing
		Rhythm-Based Actions
		Dexterity-Based Actions
		Memorizing
		Puzzle Solving
		Luck

	Planning	Tradeoffs
		Randomness
		Risk/Reward
		Predictable Consequences
		Limited Planning Ability
		Strategic Knowledge
		Stimulated Planning
	Balancing	Balancing Effects
		Symmetry
		Team Balance
		Riht Level of Difficulty
		Right Level of Complexity
		Handicaps
		Paper-Rock-Scissors
Meta Games, Replayability and Learning Curves	Meta Games	Meta Games
	Replayability and Learning Curves	Replayability
		Varied Gameplay
		Smooth Learning Curves

Appendix B

Complete Survey of GDPs

Table B.1: Game Design Patterns - Full Research

		Games / GDPs	Sim SE	SE RPG	PlayScrum	SESAM	AMEISE	SimJavaSP	iTest Learning	XMED	A Ilha dos Requisitos	SimSoft	O Jogo das 7 Falhas	iLearn Test	CRobots	Total of GDPs
Game Elements	Game Worlds	Game World	X	X				X			X			X	X	6
		Reconfigurable Game World														0
		Levels		X					X	X	X	X	X	X		7
		Inaccessible Areas									X					1
		Consistent Reality Logic	X	X		X	X	X	X	X	X	X	X	X	X	12
		Alternative Reality									X					1
		Moveable Tiles														0
	Objects	Enemies	X					X					X		X	4
		Boss Monsters														0
		Deadly Traps														0
		Obstacles			X						X			X	X	4

		Avatars		X										X	X	3
		Units	X	X		X	X	X								5
		Tools	X					X			X				X	4
		Controllers												X		1
		Alarms	X													1
		Pick-Ups														0
		Power-Ups														0
		Clues	X	X		X	X	X	X	X	X	X	X	X		11
		Extra-Game Information									X	X				2
	Abstract Objects	Score	X	X		X	X	X	X	X	X	X	X	X		11
		High Score Lists					X		X		X					3
		Lives					X				X				X	3
	Locations	Strategic Locations														0
		Outstanding Features														0
		Chargers														0
Resource and	Types of Resources	Resources	X	X	X	X	X	X			X					7
Resource Management	Resource Control	Producer-Consumer														0
		Ownership			X											1
		Resource Management	X	X	X	X	X	X			X					7
	Resource Control	Investments	X		X											2
		Diminishing Returns						X								1
Information, Communication	Information Quality	Imperfect Information	X	X	X	X	X	X			X		X			8
and Presentation		Perfect Information							X	X		X		X	X	5
		Uncertainty of Information						X								1
	Information Distribution	Symmetric Information													X	1
		Asymmetric Information			X											1
		Public Information													X	1
	Information Access	Communication Channels			X										X	2
	Information Presentation	Game State Overview	X	X	X	X	X	X	X	X	X	X	X	X	X	13
Actions and Events	Actions	Combat													X	1
		Movement									X			X	X	3
		Maneuvering												X		1
		Aim & Shoot														0
		Construction														0
	Action Control	Priviledged Abilities			X											1
		Asymmetric Abilities	X	X	X	X	X	X								6

		Limited Set of Actions	X	X	X	X	X	X	X	X	X	X	X	X	X	13
		Downtime			X	X	X								X	4
		Experimenting	X	X		X	X	X			X		X	X	X	9
		Trasnfer of Control			X											1
		Interruptible Actions						X								1
		Focus Loci		X										X		2
		New Abilities									X					1
		Improved Abilities														0
		Ability Losses														0
		Decreased Abilities	X					X								2
		Extended Actions														0
		Irreversible Actions	X	X					X	X	X	X				6
		Save-Load Cycles					X									1
	Rewards and Penalties	Rewards			X				X	X		X	X	X		6
		Penalties	X	X	X	X	X	X	X	X	X	X	X	X		12
		Illusionary Rewards														0
	Events	Ultra-Powerful Events			X			X								2
Narrative Structures,	Evaluation	Delayed Effects			X										X	2
Predictability and Immersion		Hovering Closures									X		X			2
		Illusion of Influence	X	X	X	X	X	X			X		X	X	X	10
		Perceived Chance to Succeed	X	X	X	X	X	X	X	X	X	X	X	X	X	13
	Immersion	Immersion														0
		Anticipation														0
	Creative Control	Freedom of Choice	X	X	X	X	X	X	X			X	X	X	X	11
		Creative Control														0
		Storytelling									X					1
	Narrative Structires	Narrative Structures									X					1
		Tension			X											1
		Characters	X	X		X	X	X		X	X					7
		Character Development									X					1
		Planned Character Development														0
		Identification														0
		Higher-level Closure as Gameplay Progresses														0
		Surprises	X			X	X	X								4
		Cut Scenes									X					1
		Easter Eggs														0

Social Interaction	Competition	Competition			X		X								X	3
		Conflict			X		X								X	3
		Player Killing													X	1
		Betrayal														0
	Collaboration	Cooperation														0
	Group Activities	Team Play														0
		Alliances														0
		Roleplaying														0
		Contructive Play														0
		Player Decided Results														0
	Stimulated Social Interaction	Social Interaction			X										X	2
		Trading														0
		Bidding														0
		Bluffing														0
		Negotiation														0
		Social Dilemmas			X											1
Goals	Goals of Ownership and Overcoming Opposition	Gain Ownership														0
		Overcome			X								X			2
		Stealth														0
		Eliminate														0
		Rescue														0
		Capture														0
		Evade														0
		Conceal			X											1
		Race			X						X		X		X	4
	Goals of Arrangement	Collection	X	X	X	X	X	X			X		X	X		9
	Goals of Persistence	Guard														0
		Survive													X	1
		Traverse														0
	Goals of Information and Knowledge	Gain Information	X		X	X	X			X			X			6
		Gain Competence														0
		Exploration									X			X		2
Goal Structures	Goal Characteristics	Predefined Goals	X	X	X	X	X	X	X	X	X	X	X	X	X	13
		Dynamic Goal Characteristics	X					X								2
		Optional Goals														0
		Interferable Goals			X	X	X								X	4

		Player Defined Goals														0
	Relations between Goals	Preventing Goals			X										X	2
		Hierarchy of Goals			X								X	X		3
		Tournaments														0
		Incompatible Goals													X	1
		Selectable Sets of Goals														0
		Supporting Goals														0
	Relations between Goals and Players	Symmetric Goals			X		X								X	3
		Asymmetric Goals														0
		Commited Goals														0
Game Sessions	Game and Play Sessions	Real-Time Games									X		X	X	X	4
		Asynchronous Games													X	1
		Syncronous Games	X	X	X	X	X	X	X	X	X	X	X	X		12
		Single-Player Games	X	X		X		X	X	X	X	X	X	X		10
		Multiplayer Games			X		X								X	3
		Turn-based Games	X	X	X	X	X	X								6
		Closure Points							X	X	X	X	X	X	X	7
	Player Activity	Player Elimination													X	1
		Analysis Paralysis														0
		The Show Must Go On														0
		Agents														0
Game Mastery and Balancing	Game Mastery	Game Mastery	X	X	X	X	X	X	X		X	X	X	X	X	12
		Empowerment	X	X	X	X	X	X	X				X	X	X	10
		Timing				X	X						X	X		4
		Rhythm-Based Actions												X		1
		Dexterity-Based Actions												X		1
		Memorizing			X											1
		Puzzle Solving														0
		Luck			X			X					X			3
	Planning	Tradeoffs														0
		Randomness	X		X			X					X			4
		Risk/Reward														0
		Predictable Consequences							X			X	X			3
		Limited Planning Ability			X					X	X	X	X			5
		Strategic Knowledge	X	X		X	X	X		X		X	X		X	9
		Stimulated Planning		X		X	X								X	4

	Balancing	Balancing Effects														0
		Symmetry													X	1
		Team Balance														0
		Right Level of Difficulty	X	X		X	X	X	X	X	X	X	X	X		11
		Right Level of Complexity	X	X		X	X	X	X	X	X	X	X	X		11
		Handicaps														0
		Paper-Rock-Scissors														0
Meta Games, Replayability	Meta Games	Meta Games														0
and Learning Curves	Replayability and Learning Curves	Replayability	X	X	X	X	X	X					X			7
		Varied Gameplay												X		1
		Smooth Learning Curves											X			1
	Total of GDP's		38	33	46	33	39	39	22	21	42	23	37	36	42	